Introduction to Artificial Intelligence, Fall & Winter 2022
College of Computer Science, Zhejiang University
Reference Solutions for Problem Set 1

丁尧相

2022 年 11 月 4 日

Problem 1.1. 若将围棋形式化为第二讲中所引入的搜索问题：
（关于围棋的简单介绍参见：http://www.homygame.com/ngscom/help/weiqi.htm）

1. 状态空间，动作集合，转移函数，初始状态，结束状态，单步代价（或奖励）各是什么?
2. 状态空间的大小是多少?
3. 若取消所有吃子和数气的规则，也就是说对弈双方可以把棋子放在棋盘任意剩余的空位处，已经放在棋盘上的棋子也不会被吃掉。此时你定义的搜索树的叶子节点有多少个?

参考解答: 本题前两问没有考虑围棋规则导致一些状态无法达到的情况。

- (1) 状态空间为棋盘上的所有局面；(2) 动作集合为在棋盘任意位置落子；(3) 转移函数是确定函数：输入为某一局面及落子，输出为对应的局面；(4) 初始状态是空棋盘；(5) 结束状态为胜负已分的局面；(6) 单步代价在非结束状态下均为 0，在结束状态下可定义输棋代价为正（奖励为 0 或负），赢棋代价为 0 或负（奖励为正，若输棋奖励为负也可以为 0）。
- 状态空间的大小为 3^{361}。
- 叶子节点为 2^{361} 个。

Problem 1.2. 对于一个搜索问题，请回答下列问题：

1. 若状态空间大小有限，是否对应的搜索树深度一定是有限的?

2. 若状态空间大小有限，且搜索问题的状态空间图是一个树，是否对应的搜索树深度一定是有限的?
3. (***) 一般地，在什么条件下，搜索树深度一定是有限的?

参考解答:

- 不是，如果状态空间图有环的话搜索树深度可以无限。
- 是，这是无环的情况。
- (1) 状态空间大小有限，且状态空间图无环；(2) 状态空间大小无限，状态空间图无环，但从初始状态到任意状态的路径长度都小于一个有限值。

Problem 1.3. (***) 第二讲中，对于 8-puzzle 问题，我们给出了两个简单的 admissible heuristic function (AHF)：曼哈顿距离，以及位置错误的方块个数。你能否自己给出一个 AHF，证明它具有 admissible 的性质，并且论述一下它和上述两个 AHF 之间的优劣?
参考解答: 总体来说能够提供越多信息的函数越好，即对于任意状态，在满足 Admissible 条件下尽量具有较大的值。对于 8-puzzle 问题，可以参考`https://cse.iitk.ac.in/users/cs365/2009/ppt/13jan_Aman.pdf`。

Problem 2.1. 考虑对双人两步零和博弈进行推广，若假设最终选定的单元格中数值为 a 时，Alice 需付出 $ka+b$ 的损失，其中 $k>0$, 而 Bob 仍然获得 a 的收益，是否仍然有后手占优势的结论? 若是，请给出证明。若否，请给出反例。
参考解答: 是，因为 Alice 最小化 $ka+b$ 等价于最小化 a。

Problem 2.2. 在双人非零和博弈中，如果除了双方都知道对方的收益函数外，收益函数没有其它限制可以任意选取，那么是否存在 α-β 剪枝完全无效的例子? 若存在，能否试着给出一个这样的博弈树?
参考解答: 是，在运气很差的情况下，搜索顺序存在最坏情况完全无法剪枝，如图 1所示。如果先进入第二个分支，则无法剪枝。(当然也可以举搜索树本身有特殊结构的例子，比如所有的叶子上的 cost 都是一样的。不过本题本意是想强调搜索顺序对于剪枝的影响。)。

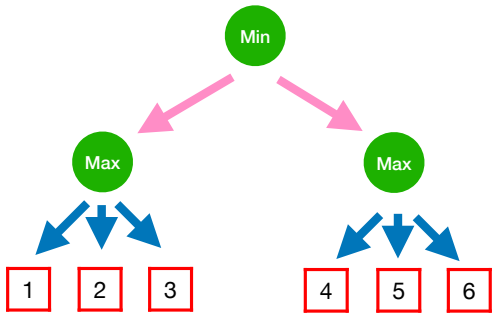

图 1: Problem 2.2 的搜索树例子。

Problem 2.3. (***) 在 MCTS 算法中，常使用完全随机的策略作为默认策略 (default policy)，这种策略看上去过于随机。我们能否同时运行一个其它搜索算法，如深度优先搜索，并在每次调用 default policy 时使用这个算法生成的结果？如果不可以，能否提出一个你认为合适的改进方法?
参考解答: 可以用其它算法生成的策略替代随机策略。实际上，AlphaGo 就使用了基于强化学习得到的策略来作为默认策略进行搜索。但要注意这里的默认策略必须十分高效，因为需要进行多次重复采样。因而深度优先搜索这样本身效率较低的策略是不合适的。

Problem 3.1. 请给出一个可以应用强化学习的实际问题的例子, 并建模其中的 MDP (Markov Decision Process)：描述出 state space, action space, transition function, reward function 各是什么。
参考解答: 按照定义给出即可。

Problem 3.2. 在图 2所示 MDP 中，表格代表了 9 个 state，单元格内的数值代表了到达这一状态能够得到的 reward，假定执行动作状态转移是确定的，MDP 中的 $\gamma = 0.9$。请分别画出使用 value iteration 和 policy iteration 前 5 轮每个 state 对应的值函数以及策略的变化情况。
提示：每一轮同样画出表格，在对应的单元格内填上值函数和当前最优策略即可。初始值函数可以全部设置为 0，初始策略可以自己指定。
参考解答: 如图 3和图 4所示。

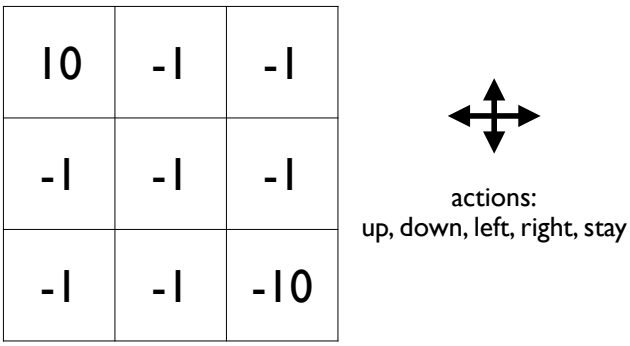

图 2: Problem 3.2 的 MDP。

Problem 3.3. (***) SARSA 和 Q-learning 都使用了 ϵ-greedy 策略进行探索，其中如何对 ϵ 进行设置是一个值得探讨的问题。请问下面几种方式是否可行？请论述你的理解。

- 在训练过程中维持一个固定的数值，如固定 $\epsilon = 0.1$。
- 在训练过程中令 ϵ 逐渐减小，但最终并不会到达 $\epsilon = 0$，而是到达一个最小数值，如 $\epsilon = 0.01$。
- 在训练过程中令 ϵ 逐渐减小，到达 0 之后再训练一些轮数，直到收敛。

参考解答: 常见做法是第二种。一般性的原则是探索所占比重在学习开始时必须很大（甚至完全探索），在学习过程中逐渐下降，并且保证始终有一定概率进行探索。但第三种在实践中未必是不可行的，需要根据实际情况来确定。

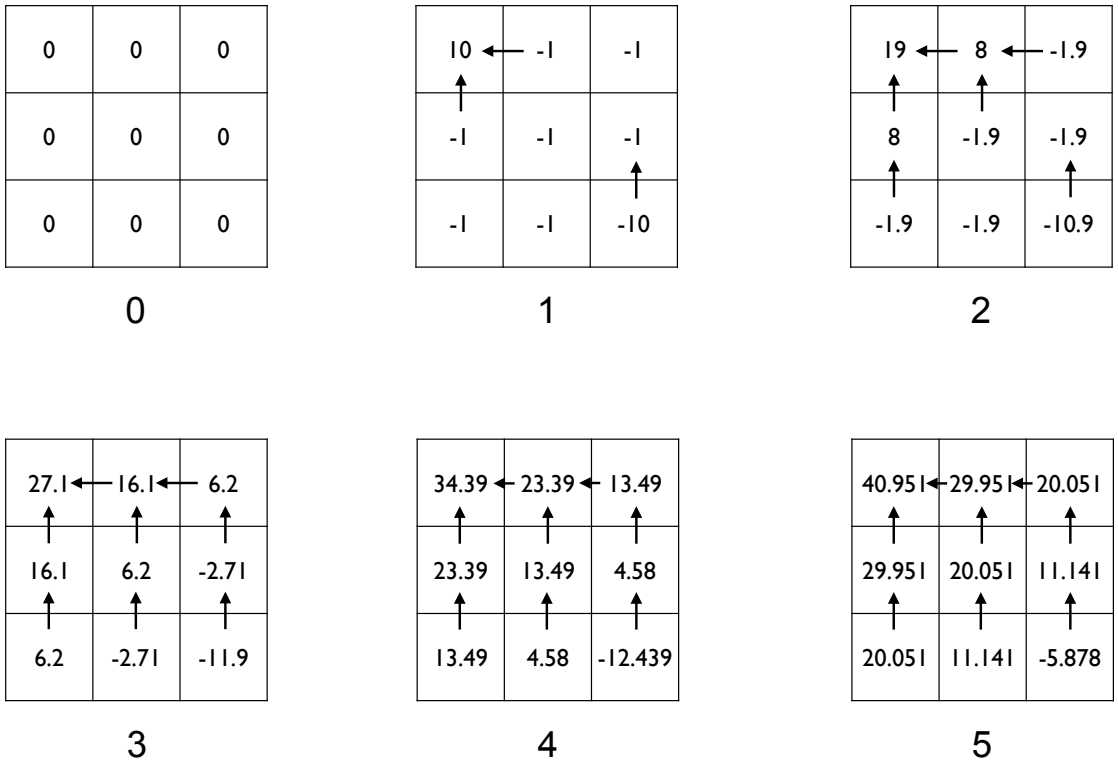

图 3: Value iteration 的结果。初始值函数设为全 0，箭头表示动作，若无箭头则表示 stay，在多个邻居值函数相同时动作为任意选取。

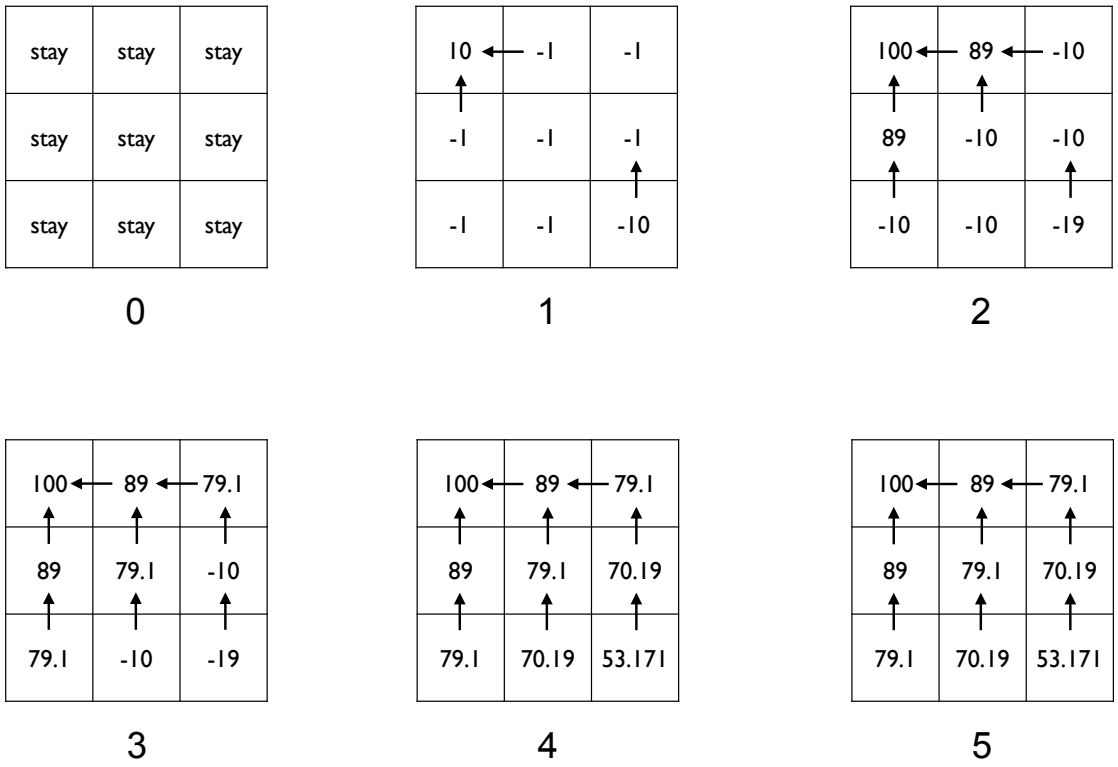

图 4: Policy iteration 的结果。